Санкт-Петербургский государственный политехнический университет

Институт информационных технологий и управления

Кафедра компьютерных систем и программных технологий

Отчёт

по лабораторной работе № 2

“Язык SQL-DDL”

по дисциплине «Базы данных»

Работу выполнил студент гр.43501/13

Михайлов Д.М. \_\_\_\_\_

Преподаватель:

Моисеев М.Ю. \_\_\_\_\_

Санкт-Петербург

2013

1. **Цель работы**

Познакомиться с основами проектирования схемы БД, языком описания сущностей и ограничений БД SQL-DDL.

1. **Программа работы**

Создание исполняемого скрипта по схеме, построенной в лабораторной работе 1.

Основная особенность при написании скрипта создания структур БД является необходимость объявления таблиц в таком порядке, чтобы все зависимости и явные ограничения были учтены. Иначе, это повлечет за собой ошибки и не все таблицы будут созданы. Другим способом решения данной проблемы может быть задание всех ограничений после полного создания структуры (генератор DDL IBExpert).

Таким образом, в данном скрипте последовательно создаются таблицы вместе с ограничениями в следующем порядке:

*Курорты, Достопримечательности, Ссылки дост.-курорты, Отели, Владение языками, Ссылки владение языками – отели, Номера,*

*Клиенты, Туры, Путевки, Билеты, Ссылки билеты – путёвки, Бронирование, Отзывы ( клиентов по курортам), Рейтинги (клиентов по отелям).*

1. **Выполнение программы работы**

**tour\_named.sql**

-- connect 'tiger.ftk.spbstu.ru:/var/lib/firebird/4081\_13/mikhailov\_db.fdb' user 'SYSDBA' password -- 'masterkey';

connect 'C:\Users\Danya\Documents\Database\tour\_named\tour\_named.fdb' user 'SYSDBA' password 'masterkey';

drop database;

commit;

create database 'C:\Users\Danya\Documents\Database\tour\_named\tour\_named.fdb' user 'SYSDBA' password 'masterkey';

commit;

connect 'C:\Users\Danya\Documents\Database\tour\_named\tour\_named.fdb' user 'SYSDBA' password 'masterkey';

CREATE TABLE RESORT (

R\_ID INTEGER NOT NULL,

R\_NAME CHAR(50),

PLACE CHAR(200),

CLIMATE CHAR(100),

WEATHER CHAR(700),

CL\_R\_RATING FLOAT);

ALTER TABLE RESORT ADD CONSTRAINT PK\_RESORT PRIMARY KEY (R\_ID);

---------------------------------------------------------------------------------------------------------------------

CREATE TABLE SIGHT (

SIGHT\_ID INTEGER NOT NULL,

SIGHTS CHAR(10000));

ALTER TABLE SIGHT ADD CONSTRAINT PK\_SIGHT PRIMARY KEY (SIGHT\_ID);

-----------------------------------------------------------------------------------------------------------------------

CREATE TABLE SIGHT\_CONNECT (

SIGHT\_RESORT\_SIGHT\_CONNECT\_ID INTEGER NOT NULL,

SIGHT\_SIGHT\_CONNECT\_ID INTEGER NOT NULL);

ALTER TABLE SIGHT\_CONNECT ADD CONSTRAINT PK\_SIGHT\_CONNECT PRIMARY KEY (SIGHT\_RESORT\_SIGHT\_CONNECT\_ID, SIGHT\_SIGHT\_CONNECT\_ID);

ALTER TABLE SIGHT\_CONNECT ADD CONSTRAINT FK\_RESORT\_CONNECT FOREIGN KEY (SIGHT\_RESORT\_SIGHT\_CONNECT\_ID) REFERENCES RESORT (R\_ID);

ALTER TABLE SIGHT\_CONNECT ADD CONSTRAINT FK\_SIGHT\_CONNECT FOREIGN KEY (SIGHT\_SIGHT\_CONNECT\_ID) REFERENCES SIGHT (SIGHT\_ID);

-----------------------------------------------------------------------------------------------------------------------

CREATE TABLE HOTEL (

H\_ID INTEGER NOT NULL,

R\_HOTEL\_ID INTEGER,

NAME CHAR(30),

PLACE CHAR(30),

STARS INTEGER,

H\_COST\_MIN INTEGER,

H\_COST\_MAX INTEGER);

ALTER TABLE HOTEL ADD CONSTRAINT PK\_HOTEL PRIMARY KEY (H\_ID);

ALTER TABLE HOTEL ADD CONSTRAINT FK\_RESORT FOREIGN KEY (R\_HOTEL\_ID) REFERENCES RESORT (R\_ID);

-----------------------------------------------------------------------------------------------------------------------

CREATE TABLE LANG (

LANG\_ID INTEGER NOT NULL,

LANGUAGE VARCHAR(50));

ALTER TABLE LANG ADD PRIMARY KEY (LANG\_ID);

-----------------------------------------------------------------------------------------------------------------------

CREATE TABLE LANG\_CONNECT (

LANG\_CONNECT\_ID INTEGER NOT NULL,

L\_CONNECT\_ID INTEGER NOT NULL);

ALTER TABLE LANG\_CONNECT ADD CONSTRAINT PK\_LANG\_CONNECT PRIMARY KEY (LANG\_CONNECT\_ID, L\_CONNECT\_ID);

ALTER TABLE LANG\_CONNECT ADD CONSTRAINT FK\_LANG\_ID FOREIGN KEY (LANG\_CONNECT\_ID) REFERENCES HOTEL (H\_ID);

ALTER TABLE LANG\_CONNECT ADD CONSTRAINT FK\_L\_ID FOREIGN KEY (L\_CONNECT\_ID) REFERENCES LANG (LANG\_ID);

-----------------------------------------------------------------------------------------------------------------------

CREATE TABLE ROOM\_INFO (

RI\_ID INTEGER NOT NULL,

H\_RI\_ID INTEGER,

RI\_ROOM\_TYPE CHAR(20),

RI\_ROOM\_COST INTEGER);

ALTER TABLE ROOM\_INFO ADD CONSTRAINT PK\_ROOM\_INFO PRIMARY KEY (RI\_ID);

ALTER TABLE ROOM\_INFO ADD CONSTRAINT FK\_RI\_HOTEL FOREIGN KEY (H\_RI\_ID) REFERENCES HOTEL (H\_ID);

-----------------------------------------------------------------------------------------------------------------------

CREATE TABLE CLIENT (

P\_ID INTEGER NOT NULL,

NAME CHAR(50),

SURNAME CHAR(50),

F\_NAME CHAR(50),

DOB DATE,

CL\_PHONE CHAR(20));

-----------------------------------------------------------------------------------------------------------------------

CREATE TABLE TOUR (

T\_ID INTEGER NOT NULL,

TOUR\_NAME CHAR(50),

WHAT\_INCLUSIVE CHAR(2000),

TOUR\_DATE DATE,

TOUR\_DURATION INTEGER,

TOUR\_COST INTEGER);

ALTER TABLE TOUR ADD CONSTRAINT PK\_CARD PRIMARY KEY (T\_ID);

-----------------------------------------------------------------------------------------------------------------------

CREATE TABLE CARD (

CARD\_ID INTEGER NOT NULL,

P\_CARD\_ID INTEGER,

T\_CARD\_ID INTEGER,

CARD\_COST INTEGER);

ALTER TABLE CARD ADD CONSTRAINT PK\_TOUR\_CONNECT PRIMARY KEY (CARD\_ID);

ALTER TABLE CARD ADD CONSTRAINT FK\_CARD\_CONNECT\_P\_ID FOREIGN KEY (P\_CARD\_ID) REFERENCES CLIENT (P\_ID);

ALTER TABLE CARD ADD CONSTRAINT FK\_CARD\_CONNECT\_T\_ID FOREIGN KEY (T\_CARD\_ID) REFERENCES TOUR (T\_ID);

-----------------------------------------------------------------------------------------------------------------------

CREATE TABLE TICKETS (

TK\_ID INTEGER NOT NULL,

TK\_NAME CHAR(100),

TK\_COST INTEGER,

TK\_TYPE CHAR(10),

TK\_DATE DATE);

ALTER TABLE TICKETS ADD CONSTRAINT PK\_TICKETS PRIMARY KEY (TK\_ID);

-----------------------------------------------------------------------------------------------------------------------

CREATE TABLE TK\_CONNECT (

T\_TK\_CONNECT\_ID INTEGER NOT NULL,

TK\_TK\_CONNECT\_ID INTEGER NOT NULL);

ALTER TABLE TK\_CONNECT ADD CONSTRAINT PK\_TK\_CONNECT PRIMARY KEY (T\_TK\_CONNECT\_ID, TK\_TK\_CONNECT\_ID);

ALTER TABLE TK\_CONNECT ADD CONSTRAINT FK\_TK FOREIGN KEY (TK\_TK\_CONNECT\_ID) REFERENCES TICKETS (TK\_ID);

ALTER TABLE TK\_CONNECT ADD CONSTRAINT FK\_CARD\_CONNECT FOREIGN KEY (T\_TK\_CONNECT\_ID) REFERENCES CARD (CARD\_ID);

-----------------------------------------------------------------------------------------------------------------------

CREATE TABLE BOOKING (

CARD\_BOOKING\_ID INTEGER NOT NULL,

RI\_BOOKING\_ID INTEGER NOT NULL,

BOOKING\_DATE DATE NOT NULL,

B\_DURATION INTEGER);

ALTER TABLE BOOKING ADD CONSTRAINT PK\_BOOKING PRIMARY KEY (CARD\_BOOKING\_ID, BOOKING\_DATE);

ALTER TABLE BOOKING ADD CONSTRAINT FK\_BK\_RI FOREIGN KEY (RI\_BOOKING\_ID) REFERENCES ROOM\_INFO (RI\_ID);

ALTER TABLE BOOKING ADD CONSTRAINT FK\_CARD\_ID FOREIGN KEY (CARD\_BOOKING\_ID) REFERENCES CARD (CARD\_ID);

-----------------------------------------------------------------------------------------------------------------------

CREATE TABLE REVIEW (

P\_REVIEW\_ID INTEGER NOT NULL,

T\_REVIEW\_ID INTEGER NOT NULL,

REV VARCHAR(1000));

ALTER TABLE REVIEW ADD CONSTRAINT PK\_REVIEW PRIMARY KEY (P\_REVIEW\_ID, T\_REVIEW\_ID);

ALTER TABLE REVIEW ADD CONSTRAINT FK\_TOUR\_CONNECT FOREIGN KEY (T\_REVIEW\_ID) REFERENCES TOUR (T\_ID);

ALTER TABLE REVIEW ADD CONSTRAINT FK\_CLIENT\_CONNECT FOREIGN KEY (P\_REVIEW\_ID) REFERENCES CLIENT (P\_ID);

-----------------------------------------------------------------------------------------------------------------------

CREATE TABLE HOTEL\_RATING (

P\_HOTEL\_RATING\_ID INTEGER NOT NULL,

H\_HOTEL\_RATING\_ID INTEGER NOT NULL,

H\_RATING INTEGER);

ALTER TABLE HOTEL\_RATING ADD CONSTRAINT PK\_H\_RATING PRIMARY KEY (P\_HOTEL\_RATING\_ID, H\_HOTEL\_RATING\_ID);

ALTER TABLE HOTEL\_RATING ADD CONSTRAINT FK\_H\_RATING FOREIGN KEY (H\_HOTEL\_RATING\_ID) REFERENCES HOTEL (H\_ID);

ALTER TABLE HOTEL\_RATING ADD CONSTRAINT FK\_P\_RATING FOREIGN KEY (P\_HOTEL\_RATING\_ID) REFERENCES CLIENT (P\_ID);

-----------------------------------------------------------------------------------------------------------------------

CREATE TABLE INSURANCE (

I\_ID INTEGER NOT NULL,

INS\_TYPE CHAR(50),

I\_HISTORY CHAR(1000),

AMOUNT INTEGER,

INS\_DATE DATE,

INS\_LENGTH INTEGER);

-----------------------------------------------------------------------------------------------------------------------

CREATE TABLE INS\_CONNECT (

P\_INS\_CONNECT\_ID INTEGER,

I\_INS\_CONNECT\_ID INTEGER NOT NULL);

ALTER TABLE INS\_CONNECT ADD CONSTRAINT PK\_INS PRIMARY KEY (I\_INS\_CONNECT\_ID);

ALTER TABLE INS\_CONNECT ADD CONSTRAINT FK\_CLIENT FOREIGN KEY (P\_INS\_CONNECT\_ID) REFERENCES CLIENT (P\_ID);

ALTER TABLE INS\_CONNECT ADD CONSTRAINT FK\_INSURANCE FOREIGN KEY (I\_INS\_CONNECT\_ID) REFERENCES INSURANCE (I\_ID);

**Также были созданы скрипты для заполнения таблиц.**

Фрагмент примера:

INSERT INTO LANG (LANG\_ID,LANGUAGE) VALUES (0,'Irish');

INSERT INTO LANG (LANG\_ID,LANGUAGE) VALUES (1,'Dutch');

INSERT INTO LANG (LANG\_ID,LANGUAGE) VALUES (2,'German');

INSERT INTO LANG (LANG\_ID,LANGUAGE) VALUES (3,'Chinese');

INSERT INTO LANG (LANG\_ID,LANGUAGE) VALUES (4,'Norway');

INSERT INTO LANG (LANG\_ID,LANGUAGE) VALUES (5,'Spain');

INSERT INTO LANG (LANG\_ID,LANGUAGE) VALUES (6,'Sweden');

INSERT INTO LANG (LANG\_ID,LANGUAGE) VALUES (7,'Norway');

INSERT INTO LANG (LANG\_ID,LANGUAGE) VALUES (8,'Russian');

INSERT INTO LANG (LANG\_ID,LANGUAGE) VALUES (9,'Chech');

INSERT INTO LANG (LANG\_ID,LANGUAGE) VALUES (10,'Russian');

INSERT INTO LANG (LANG\_ID,LANGUAGE) VALUES (11,'French');

INSERT INTO LANG (LANG\_ID,LANGUAGE) VALUES (12,'Hindi');

COMMIT;

connect 'C:\Users\Danya\Documents\Database\tour\_named\tour\_named\_5000.fdb' user 'SYSDBA' password 'masterkey';

input 'C:\Users\Danya\Documents\Database\tour\_named\1\_part\resorts\_add\_102.sql';

input 'C:\Users\Danya\Documents\Database\tour\_named\1\_part\sights\_add.sql';

input 'C:\Users\Danya\Documents\Database\tour\_named\1\_part\sights\_connect\_add\_102r.sql';

input 'C:\Users\Danya\Documents\Database\tour\_named\1\_part\hotel\_add\_102r400h.sql';

input 'C:\Users\Danya\Documents\Database\tour\_named\1\_part\lang\_add.sql';

input 'C:\Users\Danya\Documents\Database\tour\_named\1\_part\lang\_connect\_add\_400h.sql';

input 'C:\Users\Danya\Documents\Database\tour\_named\1\_part\room\_add\_400\_20.sql';

input 'C:\Users\Danya\Documents\Database\tour\_named\2\_part\clients\_add\_5000.sql';

input 'C:\Users\Danya\Documents\Database\tour\_named\2\_part\tour\_add\_2000.sql';

input 'C:\Users\Danya\Documents\Database\tour\_named\2\_part\card\_add\_7000.sql';

input 'C:\Users\Danya\Documents\Database\tour\_named\2\_part\tickets\_add\_15000.sql';

input 'C:\Users\Danya\Documents\Database\tour\_named\2\_part\tickets\_connection\_add\_7000c\_15000t.sql';

input 'C:\Users\Danya\Documents\Database\tour\_named\2\_part\booking\_add\_7000c8000r8000b.sql';

input 'C:\Users\Danya\Documents\Database\tour\_named\2\_part\review\_add\_5000cl2000t11000r.sql';

input 'C:\Users\Danya\Documents\Database\tour\_named\2\_part\rating\_add\_5000cl400h\_20000r.sql';

/\* 5000 клиентов \*/

/\* 102 курорта \*/

/\* всего 400 отелей \*/

/\* в отеле по 20 комнат \*/

/\* порядка 8000 бронь \*/

/\* 7000 кардов\*/

/\* 2000 туров \*/

/\* 15000 билетов на 7000 кардов\*/

/\* 11000 ревью \*/

/\* 20000 рейтинов \*/

1. **Выводы**

В данной работе мы познакомились с основами проектирования схемы БД, языком описания сущностей и ограничений БД SQL-DDL.

[SQL](http://ru.wikipedia.org/wiki/SQL), используемый для получения и манипулирования данными в [РСУБД](http://ru.wikipedia.org/wiki/%D0%A0%D0%A1%D0%A3%D0%91%D0%94), и сочетающий в себе элементы DDL, [DML](http://ru.wikipedia.org/wiki/DML) и [DCL](http://ru.wikipedia.org/wiki/DCL).

Функции языков DDL определяются первым словом в предложении (часто называемом запросом), которое почти всегда является глаголом. В случае с SQL эти глаголы — «[create](http://ru.wikipedia.org/wiki/Create_%28SQL%29" \o "Create (SQL))» («создать»), «[alter](http://ru.wikipedia.org/w/index.php?title=Alter_%28SQL%29&action=edit&redlink=1" \o "Alter (SQL) (страница отсутствует))» («изменить»), «[drop](http://ru.wikipedia.org/w/index.php?title=Drop_%28SQL%29&action=edit&redlink=1" \o "Drop (SQL) (страница отсутствует))» («удалить»). Это превращает природу языка в ряд обязательных утверждений (команд) к базе данных.

С помощью этого языка определяют данные, указывая тип данных , структуры представления. Результаты выполнения этих операций заносятся в системный каталог, в котором хранятся сведения о таблицах.